Count Data Regression: Negative Binomial GLMs

STAT 245

Negative Binomial Distributions

- Can be used to model count data
- Have **two parameters**:
  - a mean μ (like the Poisson "rate" λ)
  - dispersion parameter α (relates μ and variance)
- NB1: variance is linear function of μ
- NB2: variance is quadratic function of μ
- Won't detail equation, likelihood as much

Profiles in Stats: Mollie Brooks

- Developer of `glmmTMB` package
- Techn. Uni. Denmark
- Ecosystem-based marine management

NB Regression in R

```
library(glmmTMB)
theft_nb1 <- glmmTMB(Thefts ~ NEnrollment + Location
+
                       TrainingHours + SecurityCameras,
                     data = sscrime,
                     family = nbinom1(link = 'log'))

theft_nb2 <- glmmTMB(Thefts ~ NEnrollment + Location
+
                       TrainingHours + SecurityCameras,
                     data = sscrime,
```

How far off from Poisson mean = var?

```
summary(theft_nb1)
```

```
##  Family: nbinom1  ( log )
## Formula:
## Thefts ~ NEnrollment + Location + TrainingHours + SecurityCameras
## Data: sscrime
##
##      AIC      BIC   logLik deviance df.resid
##   2587.9   2619.5  -1286.0   2571.9      373
##
##
## Dispersion parameter for nbinom1 family (): 8.88
##
## Conditional model:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           2.146e+00  1.239e-01  17.322  < 2e-16 ***
## NEnrollment           1.998e-04  2.799e-05   7.140 9.32e-13 ***
## LocationTown         -1.509e-01  1.418e-01  -1.065   0.2870
## LocationUrban Fringe -3.484e-02  1.026e-01  -0.340   0.7342
## LocationRural        -2.044e-01  1.218e-01  -1.678   0.0934
```

NB1 vs. NB2

Suggestion: use scaled residual plot *or maybe* IC to choose, if no domain knowledge suggesting one

```
AIC(theft_nb1, theft_nb2)
```

```
##             df      AIC
## theft_nb1   8 2587.926
## theft_nb2   8 2557.284
```

Which fits better?

Much better than Poiss.

```
AIC(theft_pois, theft_nb1, theft_nb2)
```

```
##                df       AIC
## theft_pois      7 5265.603
## theft_nb1       8 2587.926
## theft_nb2       8 2557.284
```

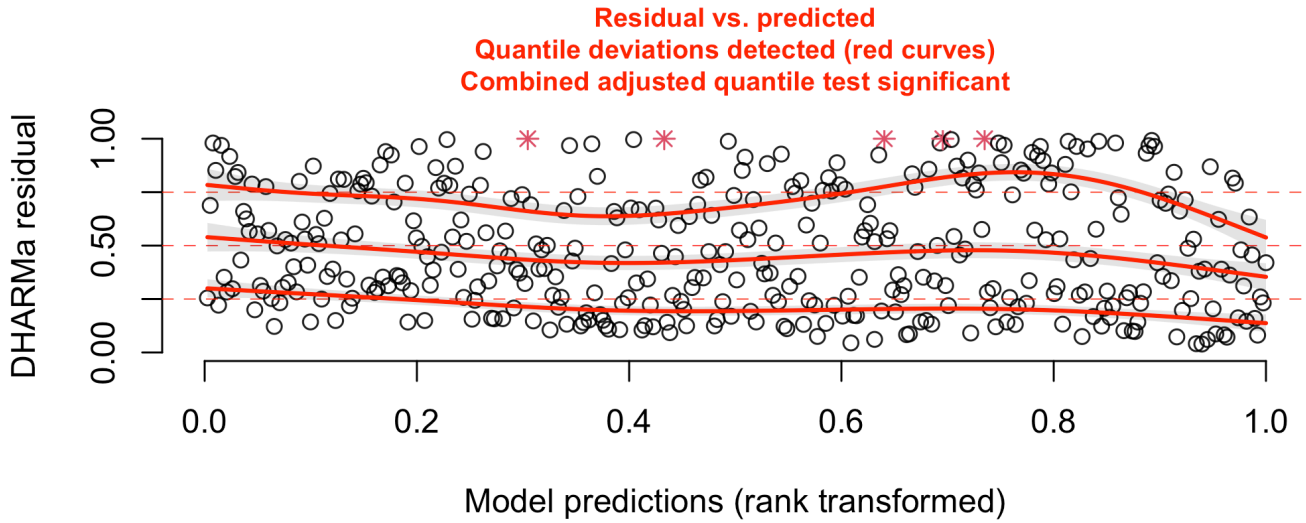
Don't use IC to compare Pois/NB;

just USE NB1 or 2, almost always.

Assessment w/DHARMa scaled residuals?

Uniform vertically -> mean-var relationship well modelled

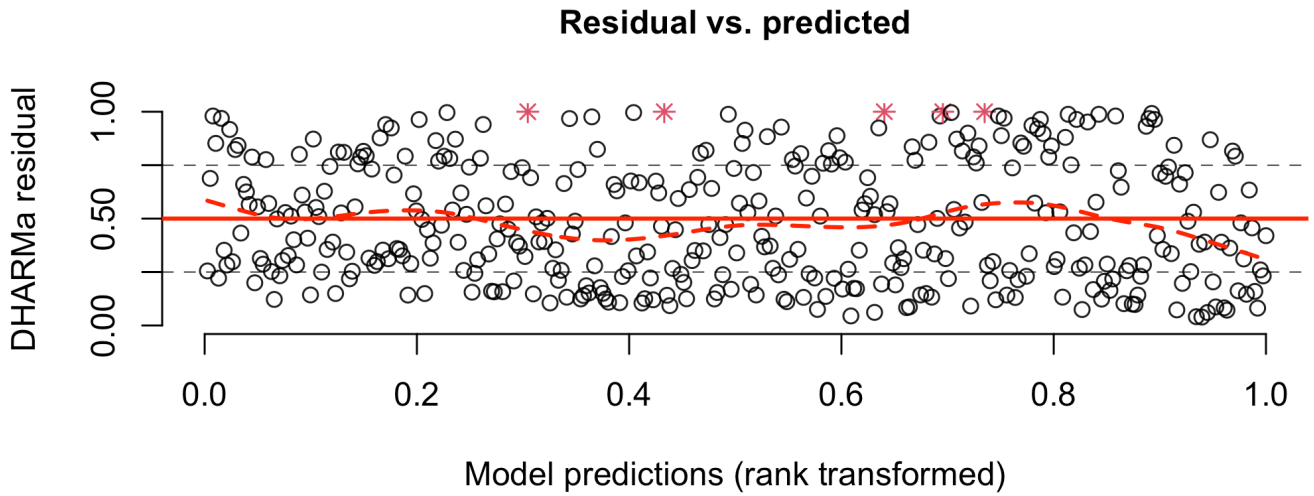
```
library(DHARMa)
nb2_sim <- simulateResiduals(theft_nb2)
plotResiduals(nb2_sim)
```



Assessment w/DHARMa scaled residuals?

Uniform vertically -> mean-var relationship well modelled

```
plotResiduals(nb2_sim,
              quantreg = FALSE)
```



Assessment w/DHARMa scaled residuals

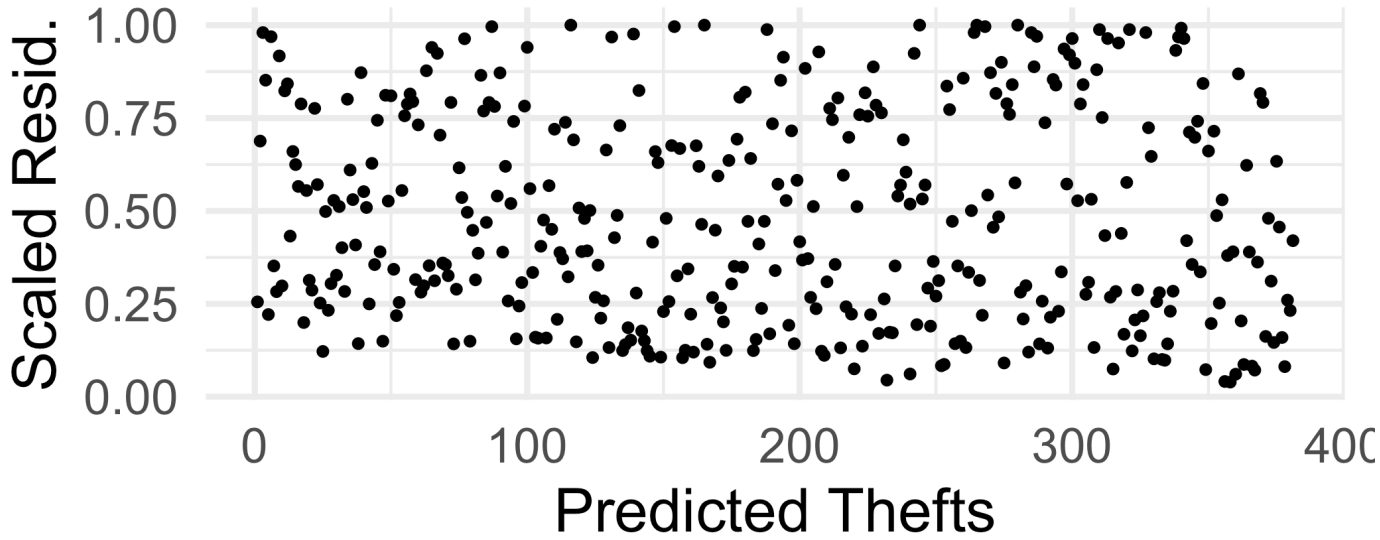
Uniform vertically -> mean-var relationship well modelled

```
sscrime <- sscrime |>
  mutate(nb2_pred =
           rank(predict(theft_nb2,
                        type = 'response')),
         nb2_scaled_resid = nb2_sim$scaledResiduals)

gf_point(nb2_scaled_resid ~
           nb2_pred,
         data = sscrime)  |>
  gf_labs(y = 'Scaled Resid.', x = 'Predicted Thefts')
```

Assessment w/scaled residuals?

Uniform vertically -> mean-var relationship well modelled



More Assessment needs doing!

- Check ACF for independence of residuals
- *No need* to check residual histogram
- Check log(response) vs. predictors and scaled residuals vs fitted and vs predictors for trends (linearity)

Next: Offsets; Model selection; interactions

Offsets

What if we need to model counts *per something*?

- Thefts per school *per student* (instead of using `NEnrollment` as predictor, essentially model `Thefts` / `NEnrollment`)
- Animal sightings *per unit effort* (sites checked; miles on trackline)

Offsets: Math

$$log(\frac{\lambda_i}{\text{effort}}) = \beta_0 + \dots$$

is the same as...

$$log(\lambda_i) = \beta_0 + \dots + log(\text{effort})$$

Offsets in R

(Use your smarts to decide if one is needed: NOT IC. Why?)

```
theft_nb2_offs <- glmmTMB(Thefts ~ Location +
                    TrainingHours + SecurityCameras +
                    offset(log(NEnrollment)),
                    data = sscrime,
                    family = nbinom2(link='log'),
                    na.action = 'na.fail')
```

Technical note: if using `dredge()` to make IC comparisons, tell `dredge()` to always include the offset term!

```
theft_nb2_offs <- update(theft_nb2_offs,
                         na.action = 'na.fail')
library(MuMIn)
dredge(theft_nb2_offs,
       rank = 'BIC',
       fixed = 'cond(offset(log(NEnrollment)))')
```

omit the `cond()` if using a Poisson GLM fitted via `glm()`